

CENG 484 - Data Mining

Assignment 3

In this assignment, you will try to implement a code that is capable of **rule extraction** from data. You will use vertebrate dataset (vertebrates.csv). You can choose Python or R for coding.

- a) Print **number of classes**, **number of attributes** and generate all possible **combinations of attributes** (twice, triple).

Examples of combination some attributes:

(Blood_Type, Give_Birth)
(Blood_Type, Can_Fly)
(Blood_Type, Live_Water)
(Blood_Type, Give_Birth, Can_Fly)
(Blood_Type, Give_Birth, Live_in_Water)
.
.
.
.

- b) Before obtain rules with multiple attributes, firstly obtain rules with **single attribute**, one attribute has to be one value in a rule such as:

Blood_Type=warm->mammals
Blood_Type=warm->birds
Blood_Type=cold->reptiles
Give_Birth=yes->mammals
Give_Birth=yes->fishes
Give_Birth=no->reptiles
.
.
.
.

c) Create rules with **two attributes** such as:

Blood_Type=warm^Give_Birth=yes->mammals

Blood_Type=warm^Give_Birth=no->birds

Blood_Type=warm^Give_Birth=no->mammals

Blood_Type=warm^Can_Fly=no->mammals

Blood_Type=warm^Can_Fly=no->birds

Blood_Type=warm^Can_Fly=yes->mammals

.

.

.

.

d) Create rules with **tree attributes** such as:

Blood_Type=warm^Give_Birth=yes^Can_Fly=no->mammals

Blood_Type=warm^Give_Birth=yes^Can_Fly=yes->mammals

Blood_Type=warm^Give_Birth=no^Can_Fly=no->birds

Blood_Type=warm^Give_Birth=yes^Live_Water=no->mammals

Blood_Type=warm^Give_Birth=yes^Live_Water=yes->mammals

Blood_Type=warm^Give_Birth=no^Live_Water=no->birds

.

.

.

.

You should **exclude** “**Name**” attribute while generating rules from the data. It is **not require** to write your code **generic** form in terms of attribute combinations, it can run rules with maximum tree attributes. The outputs of a, b, c, d, e are given as an example only, you should write your **whole output** to the report for all extracted rules from the data.

- f) Implement **coverage** and **accuracy** formula as shown in below then apply for each rule, **rank rules** according to coverage and accuracy and show **first 10 rules** with coverage and accuracy values, separately. Write your output to the report. You can review from **Chapter 5 Part-1** slides.

$$\text{Coverage}(r) = \frac{|A|}{|D|}$$

$$\text{Accuracy}(r) = \frac{|A \cap y|}{|A|},$$

where $|A|$ is the number of records that satisfy the rule antecedent,

$|A \cap y|$ is the number of records that satisfy both the antecedent and consequent,

$|D|$ is the total number of records.

- g) According to **ranked 10 rules which rule covers** which record given below, write the predicted class. At this stage **not require** to write code, check ranked rule set by eyes and write answers **to the report**. You will give answers for accuracy and coverage, separately. If a record **triggers multiple rules**, first rule will be triggered by order in rule set.

| ID | Blood_Type | Give_Birth | Can_Fly | Live_Water | Class |
|---------|------------|------------|---------|------------|-------|
| Record1 | warm | no | yes | no | ? |
| Record2 | warm | yes | no | yes | ? |
| Record3 | cold | yes | no | no | ? |
| Record4 | cold | no | no | yes | ? |

Answers by accuracy value:

Record1 -> Class = ?

Record2 -> Class = ?

Record3 -> Class = ?

Record4 -> Class = ?

Answers by coverage value:

Record1 -> Class = ?

Record2 -> Class = ?

Record3 -> Class = ?

Record4 -> Class = ?

Note: Please upload **only** Python or R **code** and your **report** (with answers) to CMS.

Please submit your solutions until **16 June 2020 23:00**. You should upload a zip file “Student_Number_Name.zip” as shown below.

Student_Number_Name.zip

|

|

|

|

|

|

|

|

|

|

|

|

|

|

|

|

|

|

|

|

|

|

|

Task_3.py or Task_3.R

Report.pdf